

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 10-124323 = EP0824236
 (43)Date of publication of application : 15.05.1998 US 664 365

(51)Int.Cl. G06F 9/445
 G06F 13/00

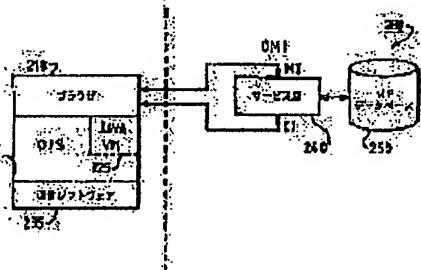
(21)Application number : 09-185402 (71)Applicant : INTERNATL BUSINESS MACH CORP <IBM>
 (22)Date of filing : 10.07.1997 (72)Inventor : RICHARD BAILD
 ALAN OGILVIE

(30)Priority
 Priority number : 96 9616649 Priority date : 08.08.1996 Priority country : GB

(54) COMPUTER SYSTEM

(57)Abstract:

PROBLEM TO BE SOLVED: To provide a mechanism which makes it possible to use information regarding a code that can be accessed through a network.
 SOLUTION: A computer work station is connected to a network such as the Internet. A browser (software) 210 runs on the computer work station can acquire data by accessing its page through the network and the page of the data is displayed on the computer work station side. A reference to a code can be added to the page of the data. The browser 201 can also obtain the code from the network and it can be executed on the computer work station. When the code is referred to, access can be gained through the network and the reference to another file containing information regarding the code such as the size, etc., is related. The browser 210 accesses the information in this file before acquiring or executing the code.



EP030016 US
 210 201

Corrs to US 664 365

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平10-124323

(43) 公開日 平成10年(1998)5月15日

(51) Int.Cl.⁶ 識別記号
G 06 F 9/445
13/00 351

F I
G 0 6 F 9/06 4 2 0 J
13/00 3 5 1 H

審査請求 未請求 請求項の数58 O.L (全 15 頁)

(21) 出願番号 特願平9-185402
(22) 出願日 平成9年(1997)7月10日
(31) 優先権主張番号 9616649.1
(32) 優先日 1996年8月8日
(33) 優先権主張国 イギリス(GB)

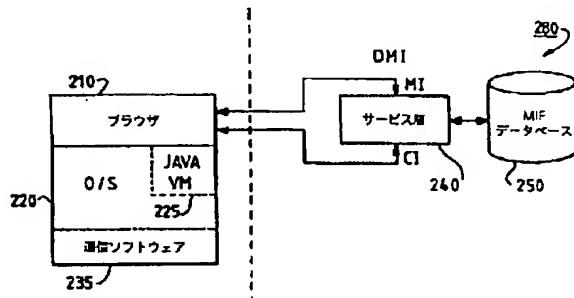
(71)出願人 390009531
インターナショナル・ビジネス・マシーンズ・コーポレイション
INTERNATIONAL BUSINESS MACHINES CORPORATION
アメリカ合衆国10504、ニューヨーク州
アーモンク (番地なし)
(72)発明者 リチャード・ペイルド
カナダ、エル3アール、8ブイ3、オンタリオ州マーカム、ウォーターブリッジ・レーン 94
(74)代理人 弁理士 坂口 博 (外1名)

(54) 【発明の名称】 コンピュータ・システム

(57) 【要約】

【課題】従って本発明は、ネットワークを通してアクセスできるコードに関する情報を使用できるようにする機構を提供する。

【解決手段】 コンピュータ・ワークステーション1がインターネット34等のネットワークに接続される。コンピュータ・ワークステーションで実行されるブラウザ（ソフトウェア）210はネットワークを通してデータのページにアクセスしてこれを取得でき、データのページはコンピュータ・ワークステーション側で表示される。データのページにはコードへの参照を追加できる。コードはブラウザもネットワークから取得してコンピュータ・ワークステーションで実行することができる。コードへの参照には、ネットワークを通してアクセスでき、コードのサイズ等、コードに関する情報を含む他のファイルへの参照も関連付けられる。ブラウザはコードを取得または実行する前にこのファイルの情報をアクセスする。



【特許請求の範囲】

【請求項1】ネットワークに接続されたコンピュータ側で実行されるよう、ネットワークを通して取得できるコードへの参照を含むデータの少なくとも1ページにアクセスできる、前記コンピュータを操作する方法であつて、

前記参照に関連付けられた、前記コードに関する情報の識別子を検出するステップと、
前記コードに関する情報を取得するステップと、
を含む、方法。

【請求項2】前記コードを実行するかどうかを決定するために、前記情報を利用するステップを含む、請求項1記載の方法。

【請求項3】前記コンピュータのユーザに前記情報の少なくとも一部を提示することで、前記コードを実行するかどうかを前記ユーザが決定できるようにするステップを含む、請求項2記載の方法。

【請求項4】前記情報の少なくとも一部を所定基準と比較することで、前記コードを実行するかどうかを決定するステップを含む、請求項2または3記載の方法。

【請求項5】前記データの少なくとも1ページはH T M L形式である、請求項1乃至4のいずれかに記載の方法。

【請求項6】前記参照はA P P L E Tタグを含む、請求項5記載の方法。

【請求項7】前記識別子は前記A P P L E Tタグ内の属性を含む、請求項6記載の方法。

【請求項8】前記識別子は、前記情報が置かれるサーバとディレクトリを指定する第1属性と、前記情報を含むファイルの名前を指定する第2属性とを含む、請求項7記載の方法。

【請求項9】前記コンピュータは、W e b ブラウザを実行して前記データの少なくとも1ページにアクセスし、前記参照は汎用リソース・ロケータ（U R L）の形である、請求項1乃至8のいずれかに記載の方法。

【請求項10】前記識別子は汎用リソース・ロケータ（U R L）の形である、請求項1乃至9のいずれかに記載の方法。

【請求項11】前記情報は、管理情報形式（M I F）に準拠した形で保存される、請求項1乃至10のいずれかに記載の方法。

【請求項12】前記情報はファイルに保存され、前記コードに関する情報を取得するステップは、前記ファイルを前記コンピュータにコピーするステップを含む、請求項1乃至11のいずれかに記載の方法。

【請求項13】前記情報は、管理情報形式（M I F）に準拠した形で保存され、前記コンピュータ側のM I Fデータベースにロードされる、請求項12記載の方法。

【請求項14】ロードされたM I F情報にアクセスするため前記M I Fデータベースに問い合わせをするステッ

プを含む、請求項13記載の方法。

【請求項15】前記情報はリモート・データベースに保存され、前記情報を取得するステップは、ネットワークを通して前記データベースにリモートで問い合わせをするステップを含む、請求項1乃至11記載の方法。

【請求項16】前記情報は前記コードのサイズを含む、請求項1乃至15のいずれかに記載の方法。

【請求項17】ネットワークを通して前記コードを取得するため前記参照を使用するステップを含む、請求項1乃至16のいずれかに記載の方法。

【請求項18】取得されたコードを前記コンピュータ側で実行するステップを含む、請求項17記載の方法。

【請求項19】ネットワークに接続でき、コンピュータ側で実行するため、前記ネットワークを通して取得できるコードへの参照を含むデータの少なくとも1ページにアクセスするコンピュータ・システムであつて、
前記参照に関連付けられた、前記コードに関する情報の識別子を検出する手段と、
前記コードに関する情報を取得する手段と、
を含む、コンピュータ・システム。

【請求項20】前記コードを実行するかどうかを決定するために、前記情報を利用する手段を含む、請求項19記載のコンピュータ・システム。

【請求項21】前記コンピュータのユーザに前記情報の少なくとも一部を提示することで、前記コードを実行するかどうかを前記ユーザが決定できるようにする手段を含む、請求項20記載のコンピュータ・システム。

【請求項22】前記情報の少なくとも一部を所定基準と比較することで、前記コードを実行するかどうかを決定する手段を含む、請求項20または21記載のコンピュータ・システム。

【請求項23】前記データの少なくとも1ページはH T M L形式である、請求項19乃至22のいずれかに記載のコンピュータ・システム。

【請求項24】前記参照はA P P L E Tタグを含む、請求項23記載のコンピュータ・システム。

【請求項25】前記識別子は、前記A P P L E Tタグ内の属性を含む、請求項24記載のコンピュータ・システム。

【請求項26】前記識別子は、前記情報が置かれるサーバとディレクトリを指定する第1属性と、前記情報を含むファイルの名前を指定する第2属性とを含む、請求項25記載のコンピュータ・システム。

【請求項27】前記コンピュータは、W e b ブラウザを実行して前記データの少なくとも1ページにアクセスし、前記参照は汎用リソース・ロケータ（U R L）の形である、請求項19乃至26のいずれかに記載のコンピュータ・システム。

【請求項28】前記識別子は汎用リソース・ロケータ（U R L）の形である、請求項19乃至27のいずれか

に記載のコンピュータ・システム。

【請求項29】前記情報は、管理情報形式（MIF）に準拠した形で保存される、請求項19乃至28のいずれかに記載のコンピュータ・システム。

【請求項30】前記情報はファイルに保存され、前記コードに関する情報を取得する手段は、前記ファイルを前記コンピュータにコピーする手段を含む、請求項19乃至29のいずれかに記載のコンピュータ・システム。

【請求項31】前記情報は、管理情報形式（MIF）に準拠した形で保存され、前記コンピュータ側のMIFデータベースにロードされる、請求項30記載のコンピュータ・システム。

【請求項32】ロードされたMIF情報にアクセスするため前記MIFデータベースに問い合わせをする手段を含む、請求項31記載のコンピュータ・システム。

【請求項33】前記情報はリモート・データベースに保存され、前記情報を取得する手段は、ネットワークを通して前記データベースにリモートで問い合わせをする手段を含む、請求項19乃至29のいずれかに記載のコンピュータ・システム。

【請求項34】前記情報は前記コードのサイズを含む、請求項19乃至33のいずれかに記載のコンピュータ・システム。

【請求項35】ネットワークを通して前記コードを取得するため前記参照を使用する手段を含む、請求項19乃至34のいずれかに記載のコンピュータ・システム。

【請求項36】取得されたコードを前記コンピュータ側で実行する手段を含む、請求項35記載のコンピュータ・システム。

【請求項37】ネットワークに接続されたコンピュータによって実行され、前記コンピュータ側で実行するためネットワークを通して取得できるコードへの参照を含むデータの少なくとも1ページにアクセスする、コンピュータ・プログラム・プロダクトであって、前記参照に連付けられた、前記コードに関する情報の識別子を検出する手段と、

前記コードに関する情報を取得する手段と、を含む、コンピュータ・プログラム・プロダクト。

【請求項38】前記コードを実行するかどうかを決定するため、前記情報を利用する手段を含む、請求項37記載のコンピュータ・プログラム・プロダクト。

【請求項39】前記コンピュータのユーザに前記情報の少なくとも一部を提示することで、前記コードを実行するかどうかを前記ユーザが決定できるようにする手段を含む、請求項38記載のコンピュータ・プログラム・プロダクト。

【請求項40】前記情報の少なくとも一部を所定基準と比較することで、前記コードを実行するかどうかを決定する手段を含む、請求項38または39記載のコンピュータ・プログラム・プロダクト。

【請求項41】前記情報はファイルに保存され、前記コードに関する情報を取得する手段は、前記ファイルを前記コンピュータにコピーする手段を含む、請求項37乃至40のいずれかに記載のコンピュータ・プログラム・プロダクト。

【請求項42】前記情報は、管理情報形式（MIF）に準拠した形で保存され、前記コンピュータ側のMIFデータベースにロードされる、請求項41記載のコンピュータ・プログラム・プロダクト。

【請求項43】ロードされたMIF情報にアクセスするため前記MIFデータベースに問い合わせをする手段を含む、請求項42記載のコンピュータ・プログラム・プロダクト。

【請求項44】前記情報はリモート・データベースに保存され、前記情報を取得する手段は、ネットワークを通して前記データベースにリモートで問い合わせをする手段を含む請求項37乃至40のいずれかに記載のコンピュータ・プログラム・プロダクト。

【請求項45】ネットワークを通して前記コードを取得するため前記参照を使用する手段を含む、請求項37乃至44のいずれかに記載のコンピュータ・プログラム・プロダクト。

【請求項46】取得されたコードを前記コンピュータ側で実行する手段を含む、請求項45記載のコンピュータ・プログラム・プロダクト。

【請求項47】インターネットを通して取得でき、コンピュータ側で実行されるアプレット・コードへの参照を含む、ワールド・ワイド・ウェブ上のデータの少なくとも1ページにアクセスするWebブラウザであって、前記参照に連付けられた、前記アプレット・コードに関する情報の識別子を検出する手段と、前記コードに関する情報を取得する手段と、を含む、ブラウザ。

【請求項48】前記コードを実行するかどうかを決定するため、前記情報を利用する手段を含む、請求項47記載のブラウザ。

【請求項49】前記コンピュータのユーザに前記情報の少なくとも一部を提示することで、前記コードを実行するかどうかを前記ユーザが決定できるようにする手段を含む、請求項47記載のブラウザ。

【請求項50】前記情報の少なくとも一部を所定基準と比較することで、前記コードを実行するかどうかを決定する手段を含む、請求項48または49記載のブラウザ。

【請求項51】前記参照はAPPLETタグを含み、前記識別子は前記APPLETタグ内の属性を含む、請求項47乃至50のいずれかに記載のブラウザ。

【請求項52】前記識別子は、前記情報を置かれるサーバとディレクトリを指定する第1属性と、前記情報を含むファイルの名前を指定する第2属性とを含む、請求項

5.1記載のブラウザ。

【請求項5.3】前記情報は、管理情報形式（MIF）に準拠した形で保存される、請求項4.7乃至5.2のいずれかに記載のブラウザ。

【請求項5.4】前記情報はファイルに保存され、前記コードに関する情報を取得する手段は、前記ファイルを前記コンピュータにコピーする手段を含む、請求項4.7乃至5.3のいずれかに記載のブラウザ。

【請求項5.5】前記情報は、管理情報形式（MIF）に準拠した形で保存され、前記コンピュータ側のMIFデータベースにロードされる、請求項5.4記載のブラウザ。

【請求項5.6】ロードされたMIF情報にアクセスするため前記MIFデータベースに問い合わせをする手段を含む、請求項5.5記載のブラウザ。

【請求項5.7】前記情報はリモート・データベースに保存され、前記情報を取得する手段は、ネットワークを通して前記データベースにリモートで問い合わせをする手段を含む請求項4.7乃至5.3のいずれかに記載のブラウザ。

【請求項5.8】ネットワークを通して前記コードを取得し、取得されたコードを前記コンピュータ側で実行する手段を含む、請求項4.7乃至5.7のいずれかに記載のブラウザ。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、コンピュータ・システム及びコンピュータ・システムからネットワークを通してデータやコードにアクセスできるように、ネットワークに接続されたコンピュータ・システムを操作する方法に関する。

【0002】

【従来の技術】過去数年間、インターネット、特にインターネットの最上部に置かれる機構の1つであるWWW（ワールド・ワイド・ウェブ）が急激に拡大している。WWWは、多くの異種サーバに分散した多数の情報ページまたはファイルで構成される。各ページはURL（汎用リソース・ロケータ）で識別される。URLはサーバ・マシン及びそのマシン上の特定のファイルまたはページの両方を示す。1つのサーバに多くのページやURLが存在可能である。

【0003】クライアントがWWWを使用するときには、WebExplorer（IBMのオペレーティング・システムOS/2の一部）、NetscapeのNavigatorプログラム等、Webブラウザと呼ばれるソフトウェアを実行する。クライアントはブラウザと対話し、特定のURLを選択する。URLによりブラウザは、そのURLまたはページに対する要求をURLで識別されたサーバに送る。通常、サーバは要求に応えるため要求されたページを取得し、そのページのデータを要求側クライアントに

転送する（クライアントとサーバの対話は、HTTP（ハイパーテキスト・トランスポート・プロトコル）に従って行われる）。このページは次にクライアントの画面に表示される。また例えば特定のテーマに関係したWWWページを検索するため、クライアントによりサーバがアプリケーションを起動することもできる。

【0004】ほとんどのWWWページは、HTML（ハイパーテキスト・マークアップ言語）と呼ばれる言語に従ってフォーマットされる。従って、通常のページにはテキストと共にタグと呼ばれるフォーマット・コマンドが埋め込まれる。タグを使用してフォントの大きさ、フォントのスタイル（イタリックかボールドか等）、テキストのレイアウト方法等が制御される。Webブラウザは指定フォーマットに従ってテキストを表示するためにHTMLスクリプトを解析する。またHTMLページには、他のURLの形で画像、ビデオ・セグメント、オーディオ・ファイル等、マルチメディア・データへの参照を加えることができる。Webブラウザはこのような参照に応答するため、データを取得して表示するかまたは再生する。このようなマルチメディア・データで専用のWWWページを周囲のHTMLテキストを使用せずに作成することも可能である。

【0005】ほとんどのWWWページには、元のページと同じサーバにある必要のない、他のWWWページへの参照もある。こうした参照は、一般にはユーザが画面の特定の位置を選択することによって起動される。通常はマウス制御ボタンをクリック（ダブルクリック）する。これらの参照またはロケーションはハイパーリンクとよばれ、通常、ブラウザにより特定の形でフラグがたてられる（例えばハイパーリンクに関連付けられたテキストの色を変えることができる）。ユーザがハイパーリンクを選択すると、参照されたページが取得され、現在表示されているページに代わる。

【0006】この他、HTMLやWWWについては、Dr Dobbs Journal、December 1994のDouglas McArthurによる"World Wide Web and HTML"、及びIan Grahamによる"The HTML SourceBook"（John Wiley、New York、1995）を参照されたい。

【0007】ここまで述べたことから言えば、また現在実現されているものとして概要に言及すると、WWWにはサーバからクライアントにダウンロードされたページは基本的に受動的である。言い換えると、クライアント・マシン側で実行されるコードを持たないという欠点がある。これが意味することは、1つには、サーバはクライアントとサーバの対話（インタラクション）に関連した処理をクライアントにオフロードできないということである。従って、クライアントが例えば電話番号で書式を埋める場合、電話番号の桁数等の正規のチェックは、サーバ側で行う必要がある。その場合、まずサーバ側で処理の負荷が重くなり、次に修正すべきエラーが生じた

場合には、サーバとクライアントの間で時間のかかる余計な通信が生じる。更にサーバがクライアント側で実行されるコードをダウンロードできないことは、WWWを利用するためには開発されたアプリケーションには大きな制限になる。

【0008】最近開発されたもの、特にSun MicrosystemsのJava技術をもとにしたものは、上に述べた問題を克服することを課題にしている。Java技術は主として、i) いくらかCやC++に似た新しいプログラミング言語と、ii) 仮想マシンで構成される。基本的にJavaプログラミング言語で書かれたプログラムは、バイト・コード形式にコンパイルし、次にJava仮想マシン上で実行時に解釈することができる。Java仮想マシンはバイト・コードをその物理マシンによって実行できる命令に変換する。

【0009】Javaで書かれたプログラムは、クライアント側のJava仮想マシンで実行されるバイト・コードの形でWWWを通してダウンロードできる。このようなプログラムは"アプレット(applet)"と呼ばれる。Java技術によりWWWを通してコードをダウンロードすることには大きな利点が2つある。まず、アプレットはプラットフォームに依存しない。ただしこれは各クライアントにJava仮想マシンのコピーがあると想定する場合に限られる(クライアントのシステム側の仮想マシンは、通常はオペレーティング・システムに組み込まれているか、Webブラウザ自体に組み込まれている)。言い換えると、サーバはそのオペレーティング・システムやマシンに従って、クライアントにダウンロードするため異なるコードのバージョンを持つ必要がない。従って、1つのバージョンの関連コードだけを書き込み、維持すればよく、これはソフトウェア開発者には都合がよい。第2点として、アプレットは物理マシンではなく仮想マシンで実行されるのでセキュリティが大幅に向上升す。従って、ネットワークからコードをダウンロードするときには、クライアント側に保存されたデータやプログラムに損害を与える有害コードが(偶発的に他の形で)含まれるというリスクが常にある。しかし仮想マシンはアプレットの動作を監視できるので、そうした有害な操作を検出し防止することができる。

【0010】ソフトウェアを仮想マシンで実行するために、バイト・コードの形でサーバからクライアントにダウンロードするという考え方には、Java技術とはまた別の形でも知られていることに注意されたい(例えば米国特許番号第5347632号を参照)。

【0011】Javaアプレットを起動するため、HTMLテキストのWebページに<APPLET>タグが含まれる。このタグは、アプレットを含むURLを識別する。ブラウザはこのタグに応答するためアプレットを取得して実行する。また<PARAM>タグも定義される。これは対応する<APPLET>と</APPLET>

タグのペアの中に保持され、実行時にアプレットに渡されるパラメータを指定するのに使用できる。(APPLETタグとPARAMタグは、HTML規格に正式には取り入れられていないが、それでも多くのWebブラウザにより認識される)。Java技術やアプレットの詳細については、Laura LemayとCharles Perkinsによる"Teach Yourself Java in 21 Days" (Sams.net Publishing, Indianapolis, USA, 1996) を参照されたい。

【0012】前記のアプローチの問題は、APPLETタグ内に保持できる情報の量が制限されることである。これはまた別の問題につながることがあるが、それはHTMLページ内で参照された特定のアプレットをダウンロードするかどうかをブラウザが決定しなければならない場合である。例えばブラウザは特別に大きい(従ってネットワークで転送すると遅くなる)か、或いはクライアント側に現在インストールされているものより新しいバージョンのJava仮想マシンを必要とする、アプレットはダウンロードしようとしないことがある。

【0013】従って本発明は、ネットワークに接続されたコンピュータを操作し、前記コンピュータ側で実行するために、ネットワークを通して取得できるコードへの参照を含むデータの少なくとも1ページへのアクセスを可能にし、前記参照に関連付けられた、前記コードに関する情報の識別子を検出するステップと、コードに関する情報を取得するステップとを含むことを特徴とする方法を提供する。

【0014】従ってコンピュータは、識別子によりコードに関する情報を検出して取得することができる。この情報は次に、前記コードを実行するかどうかを決定するために使用できる。例えば情報は、好適にはコードの大きさを示し、コードが大きすぎて相応の時間内にダウンロードできないことを決定することができる。もう1つの可能性として、この情報はクライアント・ワークステーション側にはインストールできない、ビデオ、キャプチャ機構等のハードウェア支援機能がコードを実行するために必要になることを指定する。このような決定は明示的に、ユーザに提示するか、所定基準をもとに自動的に実行することもできる(或いは自動的な意志決定とユーザ主導による意志決定の両方を採用することもできる)。

【0015】この他、コードに関する情報の使用方法として考えられるのは、コードの使用にいくらかの支払いが伴う等、監視を目的にしたものや、著作権表記やライセンス契約条件の表記にアクセスできるようにする等、単に情報提供を目的としたもの等がある。更にコード全体の起動を補助することも考えられる。例えばコードが暗号化されている場合、情報によりコードを起動する前にコードの解読に必要なアルゴリズムを指定することができる。

【0016】好適な実施例では、コードに関する情報が一度取得され、必要に応じて保存されたり処理されたりした場合は、コード自体がネットワークを通して取得される。これに代えて、コードはコードに関する情報の取得より先に取得することもできる。

【0017】好適な実施例では、データの少なくとも1ページはHTML形式であり、参照はAPPLETタグを含み、識別子は前記APPLETタグ内の属性を含む。前記参照と前記識別子は、好適にはURL（汎用リソース・ロケータ）の形をとり、参照と識別子はそれぞれ2つの部分で指定される。最初の部分はサーバとディレクトリを、次の部分は実際のファイル名を指定する。この好適な実施例の場合、アプレットはクライアント・ワークステーション側のJava仮想マシンで実行するため、Javaバイト・コード形式でダウンロードされる。これは、コンピュータでWebブラウザが実行され、前記少なくとも1ページのデータ、前記コード参照及び前記情報は全てインターネットからアクセスできるという典型的な状況である。

【0018】特定のAPPLETタグは特にJava言語に関連付けられるので、今後の何らかの規格で、一般化を進めたタグ（すなわち言語依存性が少ないもの）に代わる可能性があることがわかる。もちろんこれは本発明の原理には影響を与えない。また状況によっては、前記データの少なくとも1ページはコードの参照と、これに関連する情報の識別子の他には何も含まない、或いはほとんど何も含まない可能性があることも理解されよう（言い換えると、ページの基本的目的または唯一の目的は、コードをアクセスすることとも考えられる）。

【0019】好適な実施例の場合、情報はMIF（管理情報形式）に準拠する。これは周知の規格であり、そのためコードにアクセスしようとするクライアントにより認識される可能性が高い。しかし情報はブラウザが認識できアクセスできる任意の形式で保存することも可能である。MIF情報はネットワークを通してコピーされ、好適には、コンピュータ側のMIFデータベースにロードされる。そこで情報はブラウザにより問い合わせをすることができる（或いは単に監視を目的に保存する）。これとは別に、MIF情報はリモートのMIFデータベースに保存されていてもよい。その場合、コードに関するその情報はネットワークを通して前記データベースにリモートで問い合わせをすることにより取得できる。これによりネットワークでMIFファイル全体を転送する必要がなくなる。

【0020】

【発明が解決しようとする課題】従って本発明は、ネットワークを通してアクセスできるコードに関する情報を使用できるようにする機構を提供する。本発明による方法に代わる方法としては、例えばコードに関係した情報をHTMLページ自体に用意することが考えられる。こ

れは例えば<APPLET>タグの属性として追加できる。ただし、そのためにタグがかなり大きくなる可能性があり、それだけでなく特定のアプレットを参照した各ページが、この情報のコピーを保持しなければならない結果にもなる。これは必要な記憶域の面から高コストになるだけでなく、実際的な観点から言えばメンテナンスが困難になる。アプレットの変更は複数の場所で反映しなければならないからである（確かにコードの所有者は、そのコードを参照するサイトの全てを意識することは普通ない）。もう1つ考えられるのは、コードに関する情報をコード自体の中で識別することである。そうすればコードは実行時に、最初に関連情報を与える。この方法にはしかし、コードは少なくとも部分的には常に実行されていなければならないという欠点がある。従って、ネットワークを通してアクセスできるコードに関する情報を使用できるようにする上では、本発明によって採用される方法が最も強力で柔軟性があり、効率的な方法であることがわかる。

【0021】

【課題を解決するための手段】本発明はまた、ネットワークに接続できるコンピュータ側で実行するために、ネットワークを通して取得できるコードへの参照を含むデータの少なくとも1ページにアクセスするコンピュータ・システムを提供する。コンピュータ・システムは前記参照に関連付けられた、前記コードに関する情報の識別子を検出する手段と、コードに関する情報を取得する手段とを含むことを特徴とする。

【0022】本発明はまた、コンピュータ・システム側で実行されるようネットワークを通して取得できるコードへの参照を含むデータの少なくとも1ページにアクセスするため、ネットワークに接続されたコンピュータ・システムにより実行されるコンピュータ・プログラム・プロダクトを提供する。前記コンピュータ・プログラム・プロダクトは前記参照に関連付けられた、前記コードに関する情報の識別子を検出する手段と、コードに関する情報を取得する手段とを含む。

【0023】通常、コンピュータ・プログラム・プロダクトは、ディスクケットやCD-ROMに置かれるか、またはネットワークでのダウンロードを目的にサーバ・マシンのテープやハード・ディスクに保存でき、使用時にはコンピュータ・システムの作業メモリ（通常はRAM）にコピーされる。

【0024】本発明はまた、前記コンピュータ側で実行されるよう、インターネットを通して取得できるアプレット・コードへの参照を含む、WWWのデータの少なくとも1ページにアクセスするWebブラウザを提供する。ブラウザは、前記参照に関連付けられた、前記アプレット・コードに関する情報の識別子を検出する手段と、コードに関する情報を取得する手段とを含むことを特徴とする。

【0025】

【発明の実施の形態】図1は、システム装置10、表示画面12、キーボード14及びマウス16を含むコンピュータ・システム1の概要図である。システム装置はマイクロプロセッサ(μP)22、半導体メモリ(ROM/RAM)24、ハード・ディスク・ドライブ28、及びデータが転送されるバス26を含む。ネットワーク・アダプタ・カード30は、リンク32と共にコンピュータ・ワークステーションがインターネット34とデータを交換できるようにするものである。通常、コンピュータは基本的にはハード・ディスク・ドライブから、またはネットワークからROM/RAMにロードされたソフトウェア命令をもとに、マイクロプロセッサによって制御される。図1のシステムは、IBMのパーソナル・コンピュータ、RISCSystem/6000コンピュータ等、従来のワークステーションでよい。このようなシステムについては周知の通りであり、当業者であれば、様々なシステム構成を考えることができよう。例えばCD-ROMを追加し、複数のネットワーク・アダプタ・カードを使用し、ネットワークに接続するためネットワーク・アダプタ・カードの代わりにモ뎀を使用する、オーディオ装置を加える等が可能である。こうした変形は本発明の理解には無関係である。

【0026】図2は、オペレーティング・システム220、Webブラウザ210(ブラウザはオペレーティング・システムに組み込まれている場合もある)等、シス

```

<APPLET CODEBASE = "http://ncc.hursley.ibm.com/javainfo/classes"
          CODE = "Map.class"
          WIDTH = 560
          HEIGHT = 560
>
<param name = IMAGE value = "graphics/cup-bun.jpg">
<param name = MAP value = "java.map">
<param name = TYPE value = "NCSA">
</APPLET>

```

【0028】例1：ブラウザはこのHTMLテキストを次のように解釈する。<APPLET>タグと</APPLET>タグに応答し、サーバ側の"http://ncc.hursley.ibm.com/javainfo/classes"によって指定されたディレクトリのファイル"Map.class"に置かれたコードを取得する。(HTMLでは周知の通り"</xxx>"の形のタグは、その前の対応する<xxx>タグに関連付けられた情報やデータの終わりを示す。)この処理は、基本的にはURLのWWWページのアクセスと同じである。ブラウザは次にアプレット・コンテンツ・オブジェクトを作成する。これにHEIGHTとWIDTH(属性という)の値が関連付けられる。これらは、アプレットの実行に必要な画面の量、及び特定のパラメータ値を定義する。これらのパラメータ値は、それぞれの<PARAM>タグに列挙される。<PARAM>タグは

テム1で実行されるソフトウェア要素を示す。先に述べたように、ブラウザはインターネットで情報等にアクセスし、取得したものを表示したり処理したりするためのものである。ブラウザで重要な機能はHTMLコマンドを認識する機能である。IBMのAIX、OS/2等のオペレーティング・システム220は、ワークステーションの一般的なシステム管理を担う。オペレーティング・システムには、ワークステーションがインターネット34と対話できるようにする通信ソフトウェア235が含まれる。周知の通り、このような通信のベースはTCP/IPプロトコルである。オペレーティング・システムはまたJava仮想マシン(VM)225を含む。仮想マシン225はJavaアプリケーションを実行するために、Javaバイト・コードを解釈することができる。Java仮想マシンはブラウザの一部として提供される場合もある。システム1では、実施例によってはデスクトップ管理インターフェイス(DMI)・ソフトウェア280も実行される(これもオペレーティング・システム220に組み込まれることがある)。詳しくは後述する。

【0027】システム1とそのソフトウェアはHTTPプロトコルを使用し、これまでの方法で、Javaアプレットを含めてWWWページにアクセスし処理するため使用できる。Javaアプレットを参照するWebページのHTMLテキストの一例をここに示す。

【数1】

タグの名前と値を指定し、JavaのgetParameterメソッドを使用することでアプレット自体に取り込むことができる。従って、例えばTYPEは、値"NCSA"が割り当てられるMap.classアプレットの文字列変数の名前である。最後にアプレットは、先にアプレット・コンテンツ・オブジェクトに保存されていたパラメータ・データを使用してアプレット・オブジェクトを作成してからJavaのinit()命令を使用してアプレット・オブジェクトを起動することによって実行される。

【0029】先に述べた通り、これまでの方法ではいくつか問題が生じる。例えばJavaアプレットには、クライアントにインストールされたものよりも新しいバージョンのJava仮想マシンが必要になることがある。このような状況では、クライアントはアプレットをダウンロードして実行しようとすると、アプレットの操作は

制限されている、或いは不可能であることを認識することがある。

【0030】本発明は、この問題を克服するために、アプレットに関してブラウザに与えられる情報を拡張す

```

<APPLET CODEBASE = "http://ncc.hursley.ibm.com/javainfo/classes"
          CODE = "Map.class"
          MIFBASE = "http://ncc.hursley.ibm.com/javainfo/mifs"
          MIF = "Map.mif"
          WIDTH = 560
          HEIGHT = 560
>
<param name = IMAGE value = "graphics/cup-bun.jpg">
<param name = MAP value = "java.map">
<param name = TYPE value = "NCSA">
</APPLET>

```

【0031】例2：例2は、<APPLET>タグに2つの新しい属性が追加されていることを除いて例1と同じである点に注意されたい。2つの属性、MIFBASEとMIFはそれぞれサーバ/ディレクトリとファイルを含むURLを指定する点で、フィールドCODEBASEとCODEに似ている。ただし、CODEBASEとCODEはJavaアプレットの実際のコードを検出するのに対して、MIFBASEとMIFはアプレットを記述した情報を持つファイルを検索する。このファイルはアプレット・コード自体と同じサーバに、また同じディレクトリにも置け、或いは全く別のサーバに置いてもよい。

【0032】好適な実施例の場合、アプレットを記述した情報を含むファイルは、管理情報形式（MIF）と呼ばれる特別な形式である。これはデスクトップ管理インターフェイス（DMI）の一部としてデスクトップ管理タスクフォース（DeskTop Management TaskForce）というコンソーシアムにより開発された標準形式である。デスクトップ管理タスクフォースとMIFの概要については、Daniel Begunによる"The DMI Waiting Game"（Computer Shopper、n8、v15、p608、August 1995）及びBarbara Genglerによる"The End of the Balancing Act"（Internetwork、n8、v6、p47、August 1995）を参照されたい。MIFに関する詳細はデスクトップ管理タスクフォース（米オレゴン州ヒルズバラ）やデスクトップ管理タスクフォースのWebサイト（特に次を参照）から得られる。

【数3】

http://www.dmtf.org/general_info/factsheet.html、及び

http://www.dmtf.org/techlinks/white_papers.html

【0033】簡単に説明すると、ブレーン・テキストの形のMIFファイルは複数のグループで構成される。特別な標準グループが定義されており、いくつかは必須項目であるが、ユーザが定義したグループも受け入れられ

る。従って本発明により、例1のHTMLテキストは次のように変更される。

【数2】

る。各グループの中に名前付き属性とその値のリストがある。例えば必要なグループとして"ComponentID"があり、これは製造者、製品、バージョン、シリアル番号、インストール（日付と時刻）及び確認（当該要素が実際に存在し動作するかどうか）の各属性を持つ。図2に戻ると、DMI280をサポートするシステムにはMIFデータベース250と、サービス層240がある（これらはマシンのオペレーティング・システムの一部として提供することができる）。管理可能なプロダクト（アプリケーション・ソフトウェア等）は、それぞれのMIFファイルをサービス層のコンポーネント・インターフェイス（CI）を介してMIFデータベースに追加或いはデータベースのMIFファイルを更新する。一方、管理アプリケーションはMIFデータベースにサービス層の管理インターフェイス（MI）を介して管理可能なアプリケーションに関する情報を問い合わせることができる。ソフトウェアのMIFファイルの例と資料は、次のWebサイトから入手することができる。

【数4】

<http://www-uk.hpl.hp.com/people/arp/dmtf/ver2.htm>

【0034】ある簡単な実施例での本発明の動作を図3に示している。最初、ブラウザ・ソフトウェアが<APPLET>タグの存在を検出する（ステップ310）。このタグはダウンロードされ実行されるアプレットへの参照を示す。次にブラウザは、参照されたアプレットに関連付けられたMIFBASE、MIFの各属性を探す（ステップ320）。これらが検出されたとするとブラウザは次に、指定されたMIFファイルにアクセスして取得する（ステップ330、340）。MIFファイルは次にウィンドウに表示される（ステップ350）。ここでユーザはMIF情報をスクロールでき、ダイアログ・ボックスでアプレットの操作に進むかどうか聞かれる（ステップ360）。MIF情報が受け入れ可能となるとユーザはそこでアプレットをダウンロードして起動することができる（ステップ370、380）。しかしア

アプレットが大きすぎる、或いはJava仮想マシンにはクライアントにインストールされたものより新しいバージョンが必要である等、MIFファイルに問題のあることがわかった場合、ユーザは操作を中断するオプションを選択することができる。これによりアプレットを必要にダウンロードしたり実行を試したりする事なくなる。

【0035】ステップ320に戻るが、MIF、MIFBASEの各属性が検出されない場合は、アプレットを直接ダウンロードし起動するための処理が行われる。これは、MIF、MIFBASEの各属性を認識しないブラウザによって取られるパスであり、この例は従来の技術のブラウザに見られる。

【0036】図3に示した方法は比較的に簡単であるが、ユーザがMIFファイルそのものを走査しなければならないという欠点がある。より複雑な処理方法が図4の実施例に示してある（この場合、APPLEタグにはMIF/MIFBASE属性がないことを前提している）。MIF情報がサーバから取得されると（ステップ405）、サービス層のコンポーネント・インターフェイス（CI）を介してコンピュータ・システム（410）のMIFデータベースにロードされる。次に、MIFファイルからMIF情報を読出すためにサービス層の管理インターフェイス（MI）が用いられる（ステップ415）。ブラウザとDMIの間のこの対話は図2に概要を示している。

【0037】ブラウザは、所定の基準に照らしてMIF情報を調べる（ステップ420）。これによりもう1つのオペレーティング・システム（Java仮想マシン）のバージョン番号が、クライアントにインストールされたバージョンに対応するかどうかが確認される（ブラウザはDMIを使用して、このテストには必要ない属性は無視し、MIFファイルから選択された情報だけを取得できることに注意されたい）。所定基準が満足されない場合、アプレットの処理は終了し、アプレットがダウンロードされることはない。決定のために常時または条件に応じてユーザに提示される何らかの基準があつてもよい。例えばアプレットが所定サイズより大きい場合、アプレットをダウンロードするのに必要な時間の長さを考慮して、操作を続けるかどうかユーザに確認を求めることができる（ステップ425）。所定基準が満足されたとすると（必要なユーザ確認があるとき）、アプレットがダウンロードされる（ステップ430）。この段階で通常はセキュリティのために別のテストを行うことができる（ステップ435）。例えばアプレットが正しいサイズかどうか確認でき、MIFファイルで指定された特定のバイト数が、アプレットそのものの内容と合致しているかどうかのチェックが行える。所定基準が再び満たされたとすると（ここでもステップ440で必要なユーザ確認があるかどうかによる）、アプレットを起

動することができる（ステップ445）。ブラウザは後にクライアント・システムからアプレットを削除する場合、一般にはMIFデータベースを更新する必要がある。

【0038】図3、図4に示した処理については様々な変形が可能であることは理解されよう。例えばMIFファイルは、所定基準に照らしたチェックを行うことなく、或いはユーザ確認を求めることなく、ただ監視を目的にMIFデータベースにロードすることもできる（ステップ410）。通常これは、アプレットがダウンロードされた後に行われ（ステップ430）、そこでブラウザがMIFファイルの内容とは無関係にアプレットを起動しようとする。またブラウザによっては、テストをユーザに明らかにすることなく、所定基準に対するテストを全て自動的に行うこともできる。もう1つ考えられるのは、MIFファイルはアプレットを実行するために必要な情報を保持してもよい。例えばアプレットは変更の可能性を小さくするため、暗号化された形式でダウンロードすることが検討されているが、その場合、MIFによりアプレットを解読するのに必要なアルゴリズムを指定することもできる。

【0039】MIFファイルがMIFデータベースにロードされるかどうか、そうなら、ブラウザがMIF情報をMIFファイルそのものから直接、またはMIFデータベースを使用して保持するかどうかは任意である。MIFをMIFデータベースにロードするのは監視が目的なら便利であろう。またこれによりブラウザはMIを利用して、MIFファイルからMIF情報を取得することができる。これによりブラウザがMIFファイルを解析して、様々な属性の値を取得する必要はなくなる。しかし、多数のアプレットがクライアント・マシンにダウンロードされ、実行され削除される前に、そこに一時的にしか存在しない場合、場合によっては、DMIとの対話に伴うオーバーヘッドを許容するよりも、MIFファイル自体からMIF情報を直接取得する方が望ましいかもしれない。

【0040】ここで述べたMIF属性には、コードの大きさ、目的のオペレーティング・システムのバージョン番号等、既に標準のDMIグループに存在するものがあり、他方、用いられる暗号の種類等、ユーザ定義グループにおく必要があるものもある。MIFに保持できる情報の種類については厳密な制限はないが、属性セットを好適にはほとんどの、或いは全てのブラウザが解釈できるDMI規格の一部として、予め定義しておく方が望ましいことは明らかである。またブラウザがこのセットの中の特定の属性を認識しない場合でも、可能なら関連アプレットを起動できることも望ましい（これは、ブラウザが特定のHTMLタグを認識しないが、一般には、所望のフォーマットに正確に一致はしなくとも、ページの情報を表示できる状況と多少とも似ている）。

【0041】上の実施例で、MIFファイルはクライアント側で分析するためにその全体がダウンロードされる。これに代わる方法としては、アプレットに関する情報を取得するために、遠隔地にある関連MIFファイルに問い合わせをすることである。その場合、MIFファイルはMIFデータベースに組み込まれている可能性が大きく、通常、MIFデータベースへのアクセスが与えられるDMIを検出するために属性のMIFとMIFBASEが用いられる。この方法は、基本的に図2に示したものと同じであるが、ここでDMI280はオペレーティング・システム220やブラウザ210とは別のマシン上に位置する点が異なる。

【0042】DMIに対するこのリモート・アクセスを実現するメカニズムはいくつか考えられる。例えばDMI自体で、MIを介したりモート・アクセスを直接サポートすることが求められるが、これを直接的に完全に実現することは今のところできない。その場合、処理は図4に示したようになるが、ステップ410は省略され、ステップ410でのMIF情報の取得はリモートアクセス機構を介して行われる。これに代えて、ブラウザに周知のSNMP (Simple Network Management Protocol) を使用し、DMIにリモートで問い合わせをすることができる。DMIは既にこのメカニズムをサポートしているからである (SNMPについては、Douglas E ComerとDavid L Stevensによる"Internetworking with TCP/IP, Volume2", Prentice Hall, New Jersey, USA, 1991を参照されたい)。もう1つ考えられるのは、SNMPサポートをブラウザに組み込むことを回避するもので、MIFとMIFBASEにより同じマシン上の中間プログラムをMIFデータベースとして識別することである。この中間プログラムは次に、HTTPプロトコルで、MIF情報に対する要求をブラウザから受け取る。要求では特定の所望属性が指定される。中間プログラムは、これらの要求をDMIのM1に提示してから、応答をブラウザに再びHTTPプロトコルで返す。

【0043】MIF情報を遠隔地から直接使用する方が、最初にMIF情報をクライアントにコピーするよりも効率がよい場合があり、これは特にブラウザが認識するMIF属性が比較的少ない場合である。これによりMIFファイル全体をネットワークで転送する必要がなくなるからである。もちろんこの方法ではブラウザが所望の監視情報を個別に記録する必要がある。

【0044】先の説明では、アプレット情報の識別子はMIFBASE、MIFと呼ばれる2つの要素に分けられているが、識別子は適切な名前で、適切な形で任意に表現できることは明らかである。またコードに関する情報はMIFファイルにする必要はなく、SNMP管理情報ベース (MIB) 形式等、別の形式、或いはまたユーザが定義なり指定なりした他の形式であってもよい。更に例2の識別子はAPPLETタグそのものの中に置か

れるが、別のタグを<APPLET>、</APPLET>の間に置き（すなわち<PARAM>タグと同様）、ここに識別子を保存してもよい。もちろん、前記のことを全て考慮した上で、ある1つのオプションの方へ標準化し、その後全てのブラウザ或いはほとんどのブラウザがこれをサポートできることが望ましい。

【0045】更に、上に述べた実施例では、MIFファイルを調べて解釈するコードがブラウザに組み込まれているが、このコードはクライアント側に存在し、ブラウザから認識される別のプログラムまたはオブジェクトを形成することもできる。従ってMIFファイルを取得する際、ブラウザはMIFファイルをMIFインターフリタ (プログラム) に渡す。インターフリタはそこでファイルを所望の方法で分析する。例えばインターフリタは単にMIFファイルを表示するウィンドウを開き、また可能なら操作を続けるかどうかの確認を求め (図3のステップ350、360に対応)、或いは監視を目的に単にMIFファイルをローカルのMIFデータベースに追加するか、または所定基準に照らしてMIFファイルをテストする。可能ならMIFインターフリタはブラウザに応答を返して、アプレットの実行に進むかどうかを示す。

【0046】これに代えて、MIFインターフリタは、Java仮想マシン自体から参照することもできる。仮想マシンはアプレットを実行するため起動されると、最初、MIFインターフリタを実行し、例えば監視を目的にアプレットの使用状況を記録するか、またはインストールされた仮想マシンのバージョンがアプレットを実行するのに必要なものと対応するかどうか確認する。従って、MIFインターフリタは仮想マシンで一種のプリプロセッサとして、バイト・コード・バリデータ (validator) と同様の方法で (その直前に) 働く。現在バリデータは最初に、実行のため仮想マシンに渡されたバイト・コードを確認する。この方法で仮想マシンは、好適には第3者のMIFインターフリタ・プログラムが仮想マシンに結び付けられるようにフックを持つが、同時にMIFインターフリタ機能を仮想マシンそのものに組み込むことも可能である。

【0047】付け加えると、ここまで好適な実施例について、他の場合はインターネットに接続された従来型のコンピュータ・ワークステーションを中心に説明してきたが、本発明がこれに限定されることは理解されよう。例えばコンピュータ・ワークステーションは、いわゆるネットワーク・コンピュータに置き換え、或いはネットワークを通してコードを受信できる他のハードウェア、例えばセット・トップ・ボックス、インターネットにアクセスできるテレビ等に置き換えることもできる。同様に関連するネットワークはインターネットでなくともよく、企業イントラネット (通常はHTML等のインターネット・プロトコルを使用するが、これはインター

ネット自体からは切り離されている）、或いは他のネットワークでもよい。更に好適な実施例でダウンロードされるコードは、Javaバイト・コードを含むが、このコードは任意の言語でよく、例えばJavaのように仮想マシンにより解釈される形でも、またクライアント・マシンのハードウェアで直接実行されるものでもよい。その場合、ダウンロードされるコードの目的オペレーティング・システムがクライアント・マシンにインストールされたものに一致するか確認するためにMIFファイルを使用できる。

【0048】まとめとして、本発明の構成に関して以下の事項を開示する。

【0049】(1) ネットワークに接続されたコンピュータ側で実行されるよう、ネットワークを通して取得できるコードへの参照を含むデータの少なくとも1ページにアクセスできる、前記コンピュータを操作する方法であって、前記参照に関連付けられた、前記コードに関する情報の識別子を検出するステップと、前記コードに関する情報を取得するステップと、を含む、方法。

(2) 前記コードを実行するかどうかを決定するために、前記情報を利用するステップを含む、前記(1)記載の方法。

(3) 前記コンピュータのユーザに前記情報の少なくとも一部を提示することで、前記コードを実行するかどうかを前記ユーザが決定できるようにするステップを含む、前記(2)記載の方法。

(4) 前記情報の少なくとも一部を所定基準と比較することで、前記コードを実行するかどうかを決定するステップを含む、前記(2)または(3)記載の方法。

(5) 前記データの少なくとも1ページはHTML形式である、前記(1)乃至(4)のいずれかに記載の方法。

(6) 前記参照はAPPLETタグを含む、前記(5)記載の方法。

(7) 前記識別子は前記APPLETタグ内の属性を含む、前記(6)記載の方法。

(8) 前記識別子は、前記情報が置かれるサーバとディレクトリを指定する第1属性と、前記情報を含むファイルの名前を指定する第2属性とを含む、前記(7)記載の方法。

(9) 前記コンピュータは、Webブラウザを実行して前記データの少なくとも1ページにアクセスし、前記参照は汎用リソース・ロケータ(URL)の形である、前記(1)乃至(8)のいずれかに記載の方法。

(10) 前記識別子は汎用リソース・ロケータ(URL)の形である、前記(1)乃至(9)のいずれかに記載の方法。

(11) 前記情報は、管理情報形式(MIF)に準拠した形で保存される、前記(1)乃至(10)のいずれかに記載の方法。

(12) 前記情報はファイルに保存され、前記コードに関する情報を取得するステップは、前記ファイルを前記コンピュータにコピーするステップを含む、前記(1)乃至(11)のいずれかに記載の方法。

(13) 前記情報は、管理情報形式(MIF)に準拠した形で保存され、前記コンピュータ側のMIFデータベースにロードされる、前記(12)記載の方法。

(14) ロードされたMIF情報にアクセスするため前記MIFデータベースに問い合わせをするステップを含む、前記(13)記載の方法。

(15) 前記情報はリモート・データベースに保存され、前記情報を取得するステップは、ネットワークを通して前記データベースにリモートで問い合わせをするステップを含む、前記(1)乃至(11)記載の方法。

(16) 前記情報は前記コードのサイズを含む、前記(1)乃至(15)のいずれかに記載の方法。

(17) ネットワークを通して前記コードを取得するため前記参照を使用するステップを含む、前記(1)乃至(16)のいずれかに記載の方法。

(18) 取得されたコードを前記コンピュータ側で実行するステップを含む、前記(17)記載の方法。

(19) ネットワークに接続でき、コンピュータ側で実行するため、前記ネットワークを通して取得できるコードへの参照を含むデータの少なくとも1ページにアクセスするコンピュータ・システムであって、前記参照に関連付けられた、前記コードに関する情報の識別子を検出する手段と、前記コードに関する情報を取得する手段と、を含む、コンピュータ・システム。

(20) 前記コードを実行するかどうかを決定するために、前記情報を利用する手段を含む、前記(19)記載のコンピュータ・システム。

(21) 前記コンピュータのユーザに前記情報の少なくとも一部を提示することで、前記コードを実行するかどうかを前記ユーザが決定できるようにする手段を含む、前記(20)記載のコンピュータ・システム。

(22) 前記情報の少なくとも一部を所定基準と比較することで、前記コードを実行するかどうかを決定する手段を含む、前記(20)または(21)記載のコンピュータ・システム。

(23) 前記データの少なくとも1ページはHTML形式である、前記(19)乃至(22)のいずれかに記載のコンピュータ・システム。

(24) 前記参照はAPPLETタグを含む、前記(23)記載のコンピュータ・システム。

(25) 前記識別子は、前記APPLETタグ内の属性を含む、前記(24)記載のコンピュータ・システム。

(26) 前記識別子は、前記情報が置かれるサーバとディレクトリを指定する第1属性と、前記情報を含むファイルの名前を指定する第2属性とを含む、前記(25)記載のコンピュータ・システム。

(27) 前記コンピュータは、Webブラウザを実行して前記データの少なくとも1ページにアクセスし、前記参照は汎用リソース・ロケータ(URL)の形である、前記(19)乃至(26)のいずれかに記載のコンピュータ・システム。

(28) 前記識別子は汎用リソース・ロケータ(URL)の形である、前記(19)乃至(27)のいずれかに記載のコンピュータ・システム。

(29) 前記情報は、管理情報形式(MIF)に準拠した形で保存される、前記(19)乃至(28)のいずれかに記載のコンピュータ・システム。

(30) 前記情報はファイルに保存され、前記コードに関する情報を取得する手段は、前記ファイルを前記コンピュータにコピーする手段を含む、前記(19)乃至(29)のいずれかに記載のコンピュータ・システム。

(31) 前記情報は、管理情報形式(MIF)に準拠した形で保存され、前記コンピュータ側のMIFデータベースにロードされる、前記(30)記載のコンピュータ・システム。

(32) ロードされたMIF情報にアクセスするため前記MIFデータベースに問い合わせをする手段を含む、前記(31)記載のコンピュータ・システム。

(33) 前記情報はリモート・データベースに保存され、前記情報を取得する手段は、ネットワークを通して前記データベースにリモートで問い合わせをする手段を含む、前記(19)乃至(29)のいずれかに記載のコンピュータ・システム。

(34) 前記情報は前記コードのサイズを含む、前記(19)乃至(33)のいずれかに記載のコンピュータ・システム。

(35) ネットワークを通して前記コードを取得するため前記参照を使用する手段を含む、前記(19)乃至(34)のいずれかに記載のコンピュータ・システム。

(36) 取得されたコードを前記コンピュータ側で実行する手段を含む、前記(35)記載のコンピュータ・システム。

(37) ネットワークに接続されたコンピュータによって実行され、前記コンピュータ側で実行するためにネットワークを通して取得できるコードへの参照を含むデータの少なくとも1ページにアクセスする、コンピュータ・プログラム・プロダクトであって、前記参照に関連付けられた、前記コードに関する情報の識別子を検出する手段と、前記コードに関する情報を取得する手段と、を含む、コンピュータ・プログラム・プロダクト。

(38) 前記コードを実行するかどうかを決定するために、前記情報を利用する手段を含む、前記(37)記載のコンピュータ・プログラム・プロダクト。

(39) 前記コンピュータのユーザに前記情報の少なくとも一部を提示することで、前記コードを実行するかどうかを前記ユーザが決定できるようにする手段を含む、

前記(38)記載のコンピュータ・プログラム・プロダクト。

(40) 前記情報の少なくとも一部を所定基準と比較することで、前記コードを実行するかどうかを決定する手段を含む、前記(38)または(39)記載のコンピュータ・プログラム・プロダクト。

(41) 前記情報はファイルに保存され、前記コードに関する情報を取得する手段は、前記ファイルを前記コンピュータにコピーする手段を含む、前記(37)乃至(40)のいずれかに記載のコンピュータ・プログラム・プロダクト。

(42) 前記情報は、管理情報形式(MIF)に準拠した形で保存され、前記コンピュータ側のMIFデータベースにロードされる、前記(41)記載のコンピュータ・プログラム・プロダクト。

(43) ロードされたMIF情報にアクセスするため前記MIFデータベースに問い合わせをする手段を含む、前記(42)記載のコンピュータ・プログラム・プロダクト。

(44) 前記情報はリモート・データベースに保存され、前記情報を取得する手段は、ネットワークを通して前記データベースにリモートで問い合わせをする手段を含む前記(37)乃至(40)のいずれかに記載のコンピュータ・プログラム・プロダクト。

(45) ネットワークを通して前記コードを取得するため前記参照を使用する手段を含む、前記(37)乃至(44)のいずれかに記載のコンピュータ・プログラム・プロダクト。

(46) 取得されたコードを前記コンピュータ側で実行する手段を含む、前記(45)記載のコンピュータ・プログラム・プロダクト。

(47) インターネットを通して取得でき、コンピュータ側で実行されるアプレット・コードへの参照を含む、ワールド・ワイド・ウェブ上のデータの少なくとも1ページにアクセスするWebブラウザであって、前記参照に関連付けられた、前記アプレット・コードに関する情報の識別子を検出する手段と、前記コードに関する情報を取得する手段と、を含む、ブラウザ。

(48) 前記コードを実行するかどうかを決定するため、前記情報を利用する手段を含む、前記(47)記載のブラウザ。

(49) 前記コンピュータのユーザに前記情報の少なくとも一部を提示することで、前記コードを実行するかどうかを前記ユーザが決定できるようにする手段を含む、前記(47)記載のブラウザ。

(50) 前記情報の少なくとも一部を所定基準と比較することで、前記コードを実行するかどうかを決定する手段を含む、前記(48)または(49)記載のブラウザ。

(51) 前記参照はAPPLEタグを含み、前記識別

子は前記APPLEタグ内の属性を含む、前記(47)乃至(50)のいずれかに記載のブラウザ。

(52)前記識別子は、前記情報が置かれるサーバとディレクトリを指定する第1属性と、前記情報を含むファイルの名前を指定する第2属性とを含む、前記(51)記載のブラウザ。

(53)前記情報は、管理情報形式(MIF)に準拠した形で保存される、前記(47)乃至(52)のいずれかに記載のブラウザ。

(54)前記情報はファイルに保存され、前記コードに関する情報を取得する手段は、前記ファイルを前記コンピュータにコピーする手段を含む、前記(47)乃至(53)のいずれかに記載のブラウザ。

(55)前記情報は、管理情報形式(MIF)に準拠した形で保存され、前記コンピュータ側のMIFデータベースにロードされる、前記(54)記載のブラウザ。

(56)ロードされたMIF情報にアクセスするため前記MIFデータベースに問い合わせをする手段を含む、前記(55)記載のブラウザ。

(57)前記情報はリモート・データベースに保存され、前記情報を取得する手段は、ネットワークを通して前記データベースにリモートで問い合わせをする手段を含む前記(47)乃至(53)のいずれかに記載のブラウザ。

(58)ネットワークを通して前記コードを取得し、取得されたコードを前記コンピュータ側で実行する手段を含む、前記(47)乃至(57)のいずれかに記載のブラウザ。

【図面の簡単な説明】

【図1】コンピュータ・システムの概要図である。

【図2】図1のコンピュータ・システムの主なソフトウェア要素の概要図である。

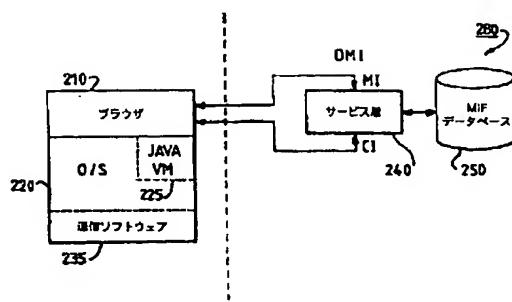
【図3】本発明の第1実施例に従った処理のフローチャートを示す図である。

【図4】本発明の第2実施例に従った処理のフローチャートを示す図である。

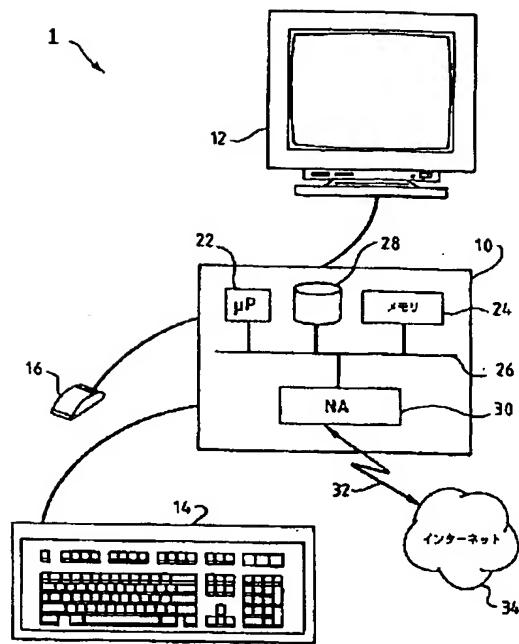
【符号の説明】

- 1 コンピュータ・システム
- 10 システム装置
- 12 表示画面
- 14 キーボード
- 16 マウス
- 22 マイクロプロセッサ
- 24 半導体メモリ (ROM/RAM)
- 26 バス
- 28 ハード・ディスク・ドライブ
- 30 ネットワーク・アダプタ・カード
- 32 リンク
- 34 インターネット
- 210 Webブラウザ
- 220 オペレーティング・システム
- 225 Java仮想マシン(VM)
- 235 通信ソフトウェア
- 240 サービス層
- 280 MIFデータベース

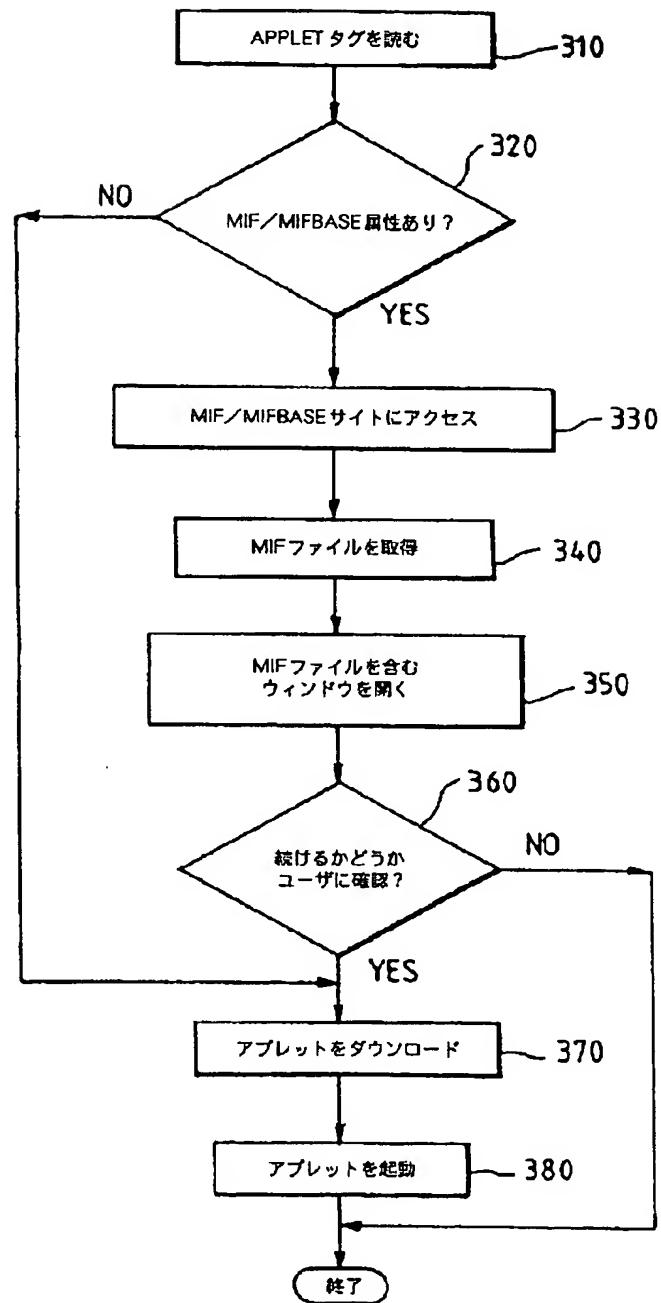
【図2】



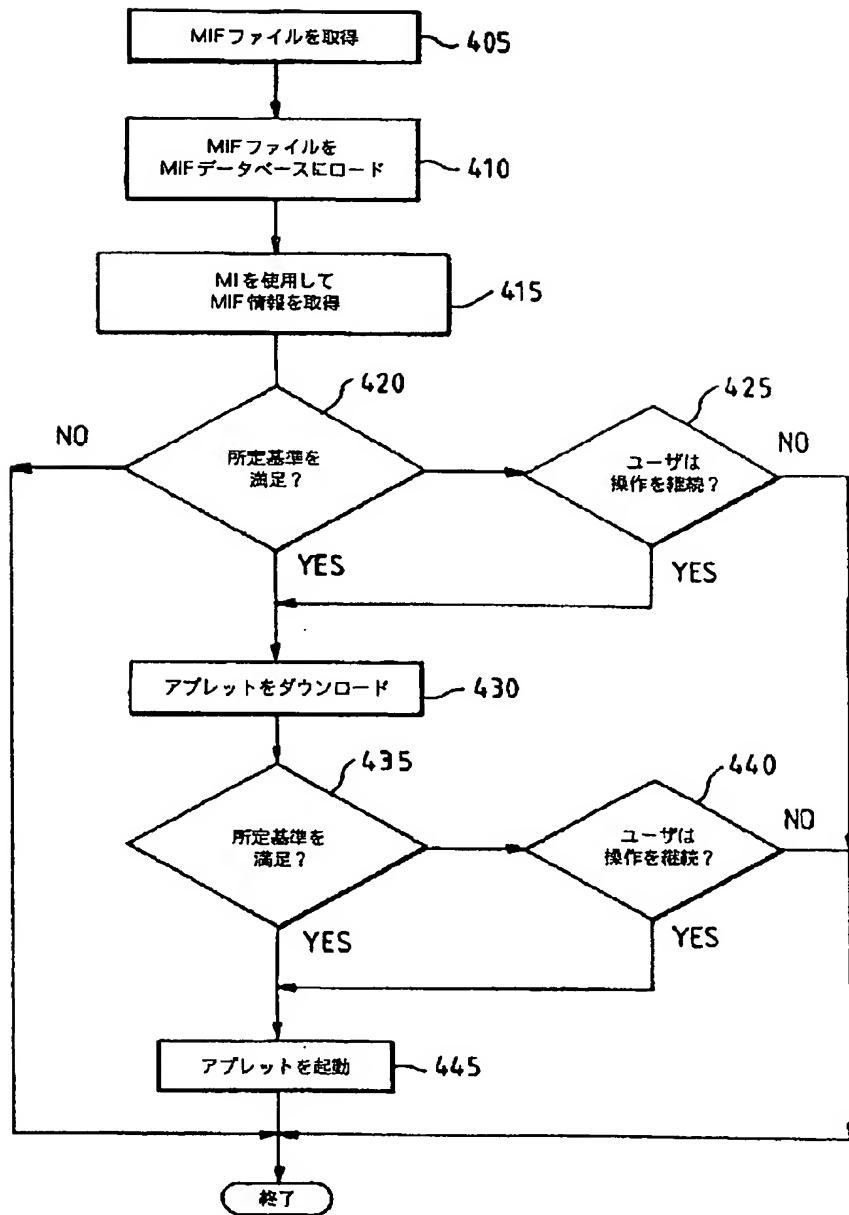
【図1】



【図3】



【図4】



フロントページの続き

(72)発明者 アラン・オギルビ
 イギリス、エス・オ-52 9エヌ・エイ
 ハンプシャー、サウスハンプトン、ノース・バッダスレイ、シルバン・ドライブ
 27